DOCUMENT RESUME

ED 280 895 TM 870 227

AUTHOR Linacre, John M.
TITLE A Computer Program for Adaptive Testing by Microcomputer.
PUB DATE Apr 87
NOTE 30p.
PUB TYPE Reports - Descriptive (141) -- Computer Programs (101)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS *Adaptive Testing; *Computer Assisted Testing; Difficulty Level; *Individual Testing; Item Analysis; Item Banks; *Latent Trait Theory; Microcomputers; Multiple Choice Tests; *Programing; *Test Format; Test Items; Test Results
IDENTIFIERS *BASIC Programing Language; Rasch Model

ABSTRACT

This paper describes a computer program in Microsoft BASIC which selects and administers test items from a small item bank. The level of the difficulty of the item selected depends on the test taker's previous response. This adaptive system is based on the Rasch model. The Rasch model uses a unit of measurement based on the logarithm of the possibilities of correctly answering a particular question. The approach used in this program is based on the developments of Rasch theory by Benjamin Wright. In this program, the questions to be asked are typed into a file. EAch question has an identifying number, the text of the question, the five possible choices, the number of the correct answer, and a preliminary estimate of the question's difficulty. Details of each test session are on another file. This file includes the name and estimated ability of the test taker, each question asked, and the responses. Through the program, the computer counts the number of correct answers and gives a numerical estimate of examinee's ability. A BASIC listing of the program to administer and score a test and six figures illustrating various operations of this computer adaptive testing process follow the study. (JAZ)

A Computer Program For Adaptive Testing By Microcomputer

John M. Linacre

Memorandum No. 40

MESA Psychometric Laboratory

Department of Education

University of Chicago

April 1987

Abstract:

A BASIC computer program is described and listed which selects and administers test items from a small item bank in a tailored manner. The program measures the test-takers' abilities and recalibrates the item bank using Rasch-based item response theory.

Keywords: Computer-Adaptive Testing, Rasch Analysis, Item Response Theory.

Article text:

Taking tests is a necessary, but often nasty, part of education. You, or any other test-taker, will probably find being tested is a more agreeable experience if the testing session is short, and the questions you are asked are targetted on your own level of ability. The computer can present this kind of test. It can quickly find out your level of knowledge by selecting, asking and scoring appropriate questions that cr ate a personalized test adapted to your performance.

Here is a program, in Microsoft BASIC, which determines your competence by choosing appropriate questions to ask you from a disk file of multiple-choice questions about a subject. When you give a right answer, the computer raises its estimate of your ability. If you give a wrong answer, its estimate of your ability is lowered. The program selects the next question to give you based on that estimate. When the estimate is accurate enough, testing stops, and you are told how you scored.

How do we measure ability?

Our first problem is how to give a numerical value to your ability. What number corresponds to easy? What number means high ability? Measuring your ability seems quite different to measuring your height.

You can find your height by using a tape measure, marked out in inches, and then measure from the floor to the top of you head.

We can do the same sort of thing for educational tests, but with an important difference. We can not measure ability directly as we can height, so we have to estimate it indirectly. We do this by observing ability in action, answering test questions right or wrong. A geography test is a method of allowing you to show your ability in geography in a way we can observe. Your answers become the means by which we measure your knowledge of geography. A computer can count up the number of correct answers and give it as your score, which is a rough indication of your ability.

How can we score an adaptive test?

When everybody is asked a different selection of questions, we can no longer make comparisons using a count of right answers alone. You may be asked the 20 easiest questions or the 10 hardest. A score of 5 on a hard test could be better than 15 on an easy test. To get round this problem. each question is given a numerical rating of its difficulty. Then the computer can give you a numerical estimate of your ability based on the difficulties of the questions you got right and wrong. You can directly compare this estimate with the ability estimates of others taking the test, no matter which questions you were asked.

You may wonder why we find an ability "estimate". In fact, no test (computerized or pencil-and-paper) can exactly determine your ability. A test can only give us a good indication of what your ability really is, in other words, an estimate.

Turning answers into measurements

Take a closer look at a geography test. We know some people are good at geography, and some are weak at it. Some geography questions are hard and some are easy. So let's imagine a vertical ruler whose scale measures knowledge of geography. Along this ruler we would like to arrange the people so that those with a poor knowledge are at the bottom, and those with a strong knowledge are at the top. Easy questions about geography, which about everyone gets right, also belong at the low end of the ruler, while hard questions, which just about everyone gets wrong, belong at the top.

What will probably happen when we give a geography test? If a knowledgeable person (shown as tall by our ruler) takes an easy item (shown as short), he will very likely get it right. If a person with little geography knowledge (shown as short) takes a difficult question (shown as tall), he will very likely get it wrong. If someone of intermediate ability in geography (mid-size on our ruler), tries a question of intermediate difficulty (mid-size too), the result will be a toss-up. Of course we cannot be sure how anyone will answer any question, we can only talk about what will probably happen.

Georg Rasch, a Danish Mathematician, imagined this scale in the 1950's. The major challenge was to make it independent of which people took which questions (Rasch, 1980). For adaptive testing, we too must have a scale that is unaffected by the fact that everyone is answering a different selection of questions from our question file.

Rasch discovered he could make a practical scale by using as his "inch" a unit of measurement based on the logarithm of the odds of someone getting the correct answer to a particular question. Let's say that we know that a particular person has a 75% chance of succeeding on a question about state capitals. This 75% probability of success means that he has 3 chances succeeding to 1 of failing, so that the scale value is the natural logarithm of 3/1 = log (3) = 1.1 logarithmic units ("logits"). A 50% chance of success, or a 50% chance of failure would be 1 chance of succeeding to 1

chance of failing, giving a scale value of the logarithm of 1/1 = log (1) = 0. Numbers like 1.1 are often difficult to think about, so in this program, we multiply these values by 9.1024, and then add 100, so that 1.1 turns into a scaled score of 110 units, and 0, the midpoint between success and failure, becomes 100 units.

When we look at test results overall, we can work out the measurement of every question in the test, and the unit measurement of everyone who has taken it, along the log odds scale, by discovering the numerical values which most closely reflect what actually happenned.

How the program uses the measurement scale

If the computer program knew your unit measurement, and the unit measurement of the question you are about to answer, it could calculate your chance of success. To start with, it does not know your measurement, so it makes a guess, and asks you a question of about that difficulty, a question for which you have apparently about a 50% chance of success. If you get it right, the program increases its estimate of your measurement, but if you get the question wrong the program decreases your measurement. Then you are asked another question targeted on your new ability measurement. Your final measurement is the place on the scale where you would probably succeed on the questions below you and fail on the questions above you. The approach used in this program is based on developments of the Rasch theory by Professor Benjamin Wright of the University of Chicago (Wright & Stone 1979; Wright & Masters 1982).

What taking a test looks like

Figure 1 shows a multiple choice question as it appears on your computer screen. The text for this screen was read from the question file. The computer selects which questions to give you. Each question has a one-line question text, and five alternative answers. You press the number on your keyboard matching your chosen answer. The computer then asks another question, until it has measured your ability accurately enough.

At the end of the test session you get a summary report like figure 2. Each line shows the identifying number of a question you took, its difficulty measurement, the answer you selected, and whether the answer was right or wrong. If your answer was quite contradictory to your estimated ability, because either you got a very easy question wrong or a very hard question right, you would see the word "SURPRISINGLY" in front of "RIGHT" or "WRONG". "SURPRISINGLY" can indicate many things: lucky guesses, careless slips, special knowledge or even mistakes in writing the questions.

What the computer is doing during the test

On starting the test, the program assumes your measurement to be near the mid-range of ability, 100 units. A question of around 100 units is asked. While you are reading the question, the computer calculates what your ability estimate would be if you got the question wrong, and also your ability if you

got it right. It then finds a question between these estimates which will be the next question it asks you. Meanwhile, you finish reading the first question, and key in the number corresponding to your choice of the correct answer. The computer checks whether this answer is correct or incorrect, and updates your ability estimate with one of the new estimates it has already calculated. It immediately displays the question already chosen to be next. You start reading this question and the computer sets about calculating possible ability levels and choosing the best question to give you next. You key in your answer once more. This process continues until the computer has calculated your ability sufficiently accurately.

At the end of the test, the computer displays a summary report of how you did. It also adds this report to the test history file on disk, which is used for reestimating the difficulty of test items. For everyone who takes your test, the computer records name, estimated ability, each question asked, answer chosen, and whether it was correct. The computer also reports whether this particular response is much as expected, right or wrong, or surprisingly right (perhaps a lucky guess), or surprisingly wrong (perhaps a careless error).

Constructing the file of questions

Frequently, the hardest part of the testing process is making up the questions and alternate answers. You will have to do this and type them into

a file in your computer. Figure 3 shows the first few questions in my file of geography questions, which I typed in using my text-editor. You can use the non-document mode of a word-processor or whatever other means you have available. You can build files for whatever topics you wish. Each file should contain questions for only one area, such as geography or math, so that you are measuring ability in only one area at a time. When you make up your questions, follow the layout in Figure 3.

Each question has 10 lines. The first line is a question identifying number for your own reference. Numbers must be in ascending order, but not every number has to be used, so that you can add or delete questions from your question file as you develop your test. The second line is the question. The third through the seventh lines each have one of the five alternative answers, one of which must be the correct one. I'm sure you've taken plenty of multiple-choice tests so that you'll know better than to have two correct answers, no correct answers or answers that do not fit in with the grammar of the question. Enter on the eigth line the number of the correct choice of answer: 1,2,3,4 or 5. "1" means the first of the five alternatives is the correct answer. If you get it wrong yourself, the program will tell you about a lot of surprisingly wrong answers when competent test-takers consistently fail to select your incorrectly specified "right" answer.

On the ninth line, you have to make an educated guess. One problem we face with a new test is that we don't know how hard the questions are. But, after

a few people have taken your test, the computer can estimate the questions' difficulties. The trouble is we need some sort of value to start with, so, the ninth line of each question contains your initial estimate of the difficulty of the question you have just written. This will be a number in the range of 1 to 200 with 100 being "average" difficulty. If you really have no idea of a particular item's difficulty, you can always enter it as 100. The computer may alter this substantially later, when you ask it to reestimate item difficulties.

Finally, leave the tenth line blank, and then key in the next question.

Put as many questions as you like in your file of test questions, upto the limit of 51 I have allowed in the program. Twenty questions is a good starting point. You can add more questions at any time, but when you add to your test, or make other changes, use new numbers, so that the program does not get confused between an old question, now deleted or changed, and a new question which happens to have the same identifying number.

Improving ability measurements - reestimating question difficulties

You can review the test history file on disk using your text editor, or "type" it to the screen, or "print" it on your computer printer. An example is shown in figure 4. It contains a complete log of each testing session.

After several people have taken your test, you can reestimate the difficulty levels of the questions and also reestimate the previous test-takers' abilities, so that they more closely correspond with the way your test is behaving overall. This is done in much the same way we estimated your ability when you took the test. Once new difficulty and ability estimates have been calculated they are included in the question and test history files and written out to disk. Figure 5 shows how the new question difficulty and its range is included before the previous difficulty estimate in the question file. Figure 6 shows how the revised ability estimate and range is included before the previous ability and range in the test history file.

Detailed explanation of the BASIC program

Now let's see how our program uses the question file and conducts a test. You can follow along in the BASIC listing in figure 8, if you like. I have included the name and purpose of each BASIC variable in figure 7. For those of you keying this program into your computer, you may leave out all program lines which contain only comments, and all comments at the end of lines. Comments start with a '. Keep the same line numbers as I have for those lines you do type in.

In lines 30 and 40, you can see that the arrays are subscripted to allow for a maximum of 51 questions and 51 test-takers. You can have more questions and people by changing all the 50's to some large number.

In line 50, the random number generator is initialized. This program uses random numbers to give everyone a different series of questions which meet the test requirements.

Line contains the function definitions to convert between report units (centered on 100) and logits, the internal unit, centered on 0.

The accuracy with which we can calculate an ability estimate is known, statistically, as the standard error of estimation. It indicates the size of the zone above and below your ability estimate in which your true ability probably lies. All measuring techniques have some sort of standard error associated with them, but it is usually not reported and so measurements are frequently thought to be more exact than they really are. In this program, the maximum standard error is set, in line 80, at about 7 units (.6 logits), so that if your ability is reported as 100 units, your true ability is probably between 93 to 107. (This program ignores the standard error of the item difficulties, which could increase the size of the probable zone by 20%.)

You then input, in line 100, the name of your file of questions.
In a subroutine starting at line 860, the 10 lines of text corresponding to each question are loaded into the question text array. A check is made to insure that that indicated correct answer on line 8 is in the range, 1 to 5, of valid options. The question difficulty on line 9 is converted from

external units to logits and checked that it is between 1 and 200 units. If an error is found the program stops and reports the approximate line number where the error was found. After successfully reading the file we return to the main program.

In line 110, you are asked the name of a file in which to store the test history. These should all be accumulated in the same file so that it can be used for test refinement later. You are then given, in line 120, the option to conduct a test. After that, in line 140, you may also reestimate the question difficulties. (You may do this as often as you like, but it only makes sense to do it after testing a number of people). If you wish to do neither, the testing session is finished.

You can use this program for giving other tests, by specifying different question and test history files.

Test Session Program Logic

A testing session starts, in line 180. You enter your name, followed by the Enter (Return) key. After this, you only need to press one key to make a response.

In line 200, the computer randomly assigns a lower limit to your ability in the range 91-95 units (-1.0 to -0.5 logits). The program uses the logit for

its internal units to simplify the math. The initial upper limit to your ability estimate is put 9 units (1 logit) higher. The initial standard error of estimation, the accuracy estimate, is assumed to be the desired accuracy level.

The computer then flags all questions as unasked in line 220, initializes the counters of questions asked and your score so far, and selects the first question. The question selection subroutine, in line 460, sets the selected question to zero. If our current standard error is better than the required accuracy, or we have asked all the questions, we immediately return without selecting another question. The question file is in its original order, so that in order not to give the same sequence of questions, we use the random number generator to give us the position in the question array from which to start looking for a question that meets our needs. We want a question that we have not asked, and whose difficulty is between our higher and lower ability estimates. This means the question is of a difficulty appropriate to our estimate of your ability. When we find a suitable question, we return. If we have been through the question array without finding one, we ask the last question we found that has not been asked.

More sophisticated techniques can be used for question selection to increase efficiency and also to check for knowledge gaps and special knowledge. The technique here is intended to minimize code and select the next question quickly.

If a question has been found, which should always happen the first time, we go to the subroutine at line 330. This displays the question and its possible answers on the screen. It updates the various arrays to show this question has been asked, but does not wait for your answer. In line 380, the computer calculates an expected score based on the previous ability estimate and the difficulty of the items encountered so far including the one just presented. The expected score is the sum of the probabilities of success on the questions. Thus, if you have the same ability estimate as an item has difficulty, i.e. your ability estimate is the same as the question's difficulty, then your probability of success is that same as your probability of failure, that is 50% or .5. So that for this question, your expected score is .5. For the test so far, we add all these probabilities together to get your expected score so far. We also get an idea of how accurately we know your ability estimate by calculating its variance, which is the sum of the product of the probability of success and the probability of failure on each question. The standard error of estimation is the reciprocal of the square root of that variance.

At this point, two actual scores are possible. You can get the question displayed on the screen right or wrong. We first assume that you get the answer wrong, so that your total count of correct answers, your score so far, will not change. In line 430, this gives a low ability estimate which, we find by adjusting our previous estimate by the difference between your actual score and what the program expected you to score, divided by the expected

score variance. However we do not want our estimate to change too quickly due to careless mistakes or other quirks and so we insure that the variance used is never less than one. We then calculate your ability if you were to get the answer to the current question right. This would give a higher estimate of your ability, by adding to the low ability estimate the value of one more right answer.

On returning to the main program at line 250, we still do not know your response to the previous question, but we assume it is going to take some time for you to read all the text that is displayed and to make a decision, so we go to line 460 to select the next question you are going to see. That question is selected by starting at a random point in the question file and selecting the first question between the high and low ability estimates, or the last question that has not yet been asked.

The next subroutine at line 240 returns to the screen to ask what you have chosen as right answer to the question you have been reading. Only a number in the range 1 to 5 is allowed. The Enter key is ignored. After receiving a selection, the computer notes it in an array and, if it is correct, updates your score and replaces the newly calculated low ability estimate with the high ability estimate. The "low" ability estimate is now our best idea of your real ability.

Back to line 250, and, if a new question has been selected, we now go

round again and display the next question. If no question has been selected, either because there are none left, or, more desirably, because our current standard error of estimation is better than the accuracy required, we fine tune our ability estimate with one more reestimation cycle in line 270. We don't increase the number of questions asked so that what before was our low estimate of ability, now becomes our most likely estimate of ability. We then tell you the test is over and display, using the subroutine at 610, the range and likely value of your ability measurement.

When the test supervisor returns to the keyboard, we display all the questions taken and results obtained so far with the subroutine at line 640. We also write this information, in even more detail, onto the test history file. The word "SURPRISINGLY" is added, in line 760, to those answers which represent an unexpected right or wrong response to a question which is more than 18 units (2 logits) harder or easier than our ability estimate.

At this point you can give another test, or reestimate question difficulties. When you first constructed the question file, you may have had to guess which are the easy and hard questions, and particularly what precise difficulty value to give them. However, after a dozen or so people have taken your test, you may want the computer to reestimate the difficulty levels of your questions based on what actually happenned. The computer will do this when you request reestimation at line 140.

In the subroutine at 960, it will read the test history file, and note for

each person what ability they were estimated to have and which questions they answered correctly and incorrectly. The reason for obtaining the test-taker's ability estimates is to give us a starting point for the reestimation procedure, and also to enable us to keep the same mean ability for the test-takers so that the reestimation procedure alters their estimates as little as possible. After reading in the responses, we ignore all test-takers and questions for whom we do not have at least one right and one wrong answer. For a big question file, it may take a good number of tests before all questions can be reestimated. At line 1250, we make sure we have enough left to be able to continue with reestimation.

At line 1270, we make an adjustment for statistical bias because we are using our data to correct our estimates. We now refine our estimates, starting in line 1340, in the same way we did when calculating abilities, but this time we change the question difficulties and test-taker abilities simultaneously. We do this through 10 cycles of reestimation which generally means that there remains no significant difference between any pair of expected and observed scores. During this procedure, at line 1480, we keep the average test-taker ability fixed so that we minimize changes in the test-takers' estimates.

After the reestimation procedure is completed at line 1550, the question file is written to disk with the new difficulty estimate, and probable range inserted at the front of the ninth line of each question. We also copy the

test history file adding, in line 1700, below each test-taker's name his revised ability estimate and its probable range. You may now continue testing, or exit the program.

References:

Rasch, G., Probabilistic Models for Some Intelligence and Attainment Tests. Chicago: University of Chicago Press, 1980

Wright, B.D. & Stone, M.H., Best Test Design: Rasch Measurement. Chicago: Mesa Press, 1979

Wright, B.D. & Masters, G.N., Rating Scale Analysis: Rasch Measurement. Chicago: Mesa Press, 1982

```
10  '  Computer-adaptive test presentation and scoring program
20  '  Copyright John Michael Linacre 1985
30 DIM PABILITY(50),PANSWER(50),PQUESTION(50),PSCORE(50),PSE(50),RESULT(50,50)
40 DIM QASKED(50),QDIFF(50),QEXP(50),QSCORE(50),QTEXT$(50,10),QVAR(50)
50 RANDOMIZE TIMER  'set random number generator so it differs every time
60  '  External units=(logits*9.1024)+100...logits=(external units-100)/9.1024
70 DEF FNU(I)=CINT(I*9.1024+100):DEF FNL(TEXT$)=(VAL(TEXT$)-100)/9.1024
80 ACCURACY=.7  'measure ability within zone of (.7*2)*.9 = 13 units
90 CLS:PRINT "Please enter control information for this test session:"
100 INPUT "What is the name of your file of questions";QFIL$:GOSUB 860
110 INPUT "What is the name of your file of test-takers";TFIL$
120 MSG$="Do you want to give a test?":GOSUB 810
130 IF YESNO$="Y" THEN GOSUB 180:GOTO 120  'Administer a test
140 MSG$="Do you want the computer to reestimate question difficulties?"
150 GOSUB 810:IF YESNO$="Y" THEN GOSUB 960:GOTO 120  'reestimate
160 PRINT "Then we have finished. Review the responses in file "+TFIL$:SYSTEM
170  '  Conduct a test session
180 CLS:PRINT "Welcome to a Computer-administered test session!":PRINT
190 INPUT "Please type your name here:",NAM$
200 ABILITY=.5*RND-1  'Starting ability is between 90 and 95
210 ABILRIGHT=ABILITY+1:SE=ACCURACY  'Upper ability estimate 100-105
220 FOR Q=1 TO QCOUNT:QASKED(Q)=0:NEXT Q  '0 means question not asked
230 PASKED=0:PRESULT=0:GOSUB 460  'select starting question
240 WHILE QSELECT<>0:GOSUB 330  'put questions and update ability estimates
250 GOSUB 460:GOSUB 540:WEND  'choose next question, check previous answer
260  '  Do another estimation to refine the measurement and finish test
270 GOSUB 380:CLS:PRINT:PRINT "You have finished your test."
280 PRINT "You";:GOSUB 610  'Display ability estimate
290 PRINT NAM$+", please call the test supervisor now."
300 MSG$="Is the test supervisor at the keyboard?":YESNO$=""
310 WHILE YESNO$<>"Y":GOSUB 810:WEND:GOSUB 640:RETURN  'Show test results
320  '  Display the question on the screen and update ability estimate
330 CLS:PRINT "Question identifier:";QTEXT$(QSELECT,1):PRINT
340 PRINT "Please select the correct answer to the following question:"
350 PRINT:PRINT QTEXT$(QSELECT,2):PRINT:PRINT "The answer is one of:":PRINT
360 FOR I=1 TO 5:PRINT I;". ";QTEXT$(QSELECT,I+2):NEXT I
370 PASKED=PASKED+1:PQUESTION(PASKED)=QSELECT:QASKED(QSELECT)=1  'This question
380 PEXP=0:PVAR=0:FOR P=1 TO PASKED  'Estimate ability based on current score
390 SUCCESS=1/(1+EXP(QDIFF(PQUESTION(P))-ABILITY))  'Probability of success
```

```
400 PEXP=PEXP+SUCCESS:PVAR=PVAR+(SUCCESS*(1-SUCCESS)):NEXT P  'sum
410 SE=SQR(1/PVAR)  'standard error of estimation = accuracy
420 IF PVAR<1 THEN PVAR=1  'limit change in estimates
430 ABILITY=ABILITY+((PRESULT-PEXP)/PVAR)  'ability so far
440 ABILRIGHT=ABILITY+(1/PVAR):RETURN  ' ability if next answer right
450  '  Select appropriate next question if needed for accuracy and available
460 QSELECT=0:IF ACCURACY>SE OR PASKED=QCOUNT THEN RETURN
470 N=INT(QCOUNT*RND)+1  'Starting point to look for suitable question
480 FOR Q=N+1 TO QCOUNT:IF QASKED(Q)=1 THEN 500 ELSE QSELECT=Q
490 I=QDIFF(Q):IF I>=ABILITY AND I=<ABILRIGHT THEN RETURN  'found one
500 NEXT Q:FOR Q=1 TO N:IF QASKED(Q)=1 THEN 520 ELSE QSELECT=Q
510 I=QDIFF(Q):IF I>=ABILITY AND I=<ABILRIGHT THEN RETURN  'found one
520 NEXT Q:RETURN  'If none are very close, default to last possibility
530  '  Get and check person's answer to question: update ability if right
540 PRINT:PRINT "Type the number of your selection here:";:LOCATE ,,1
550 TEXT$=INKEY$:WHILE TEXT$<"1" OR TEXT$>"5":TEXT$=INKEY$:WEND
560 N=VAL(TEXT$):PANSWER(PASKED)=N  'Update answer array, update score
570 I=VAL(QTEXT$(PQUESTION(PASKED),8))  'Determine correct answer
580 IF N=I THEN PRESULT=PRESULT+1:ABILITY=ABILRIGHT  'Update if correct
590 RETURN
600  '  Display estimates of ability
610 PRINT " scored in the range from";FNU(ABILITY-SE);"to";FNU(ABILITY+SE);
620 PRINT "at about";FNU(ABILITY);"after";PASKED;"questions":RETURN
630  '  Record person's ability and answers on disk
640 PRINT "Summary report on questions administered to "+NAM$
650 PRINT "Identifier","Difficulty","Answer","Right/Wrong"
660 OPEN TFIL$ FOR APPEND AS #1:PRINT #1,"Test-taker's name: "+NAM$
670 PRINT #1,"Estimated ability:";FNU(ABILITY)
680 PRINT #1,"Probable ability range:";FNU(ABILITY-SE);"-";FNU(ABILITY+SE)
690 print #1,"":FOR P=1 TO PASKED:Q=PQUESTION(P):N=PANSWER(P)
700 PRINT #1,"Question identifier:";QTEXT$(Q,1)
710 PRINT #1,"Estimated difficulty:";FNU(QDIFF(Q))
720 PRINT #1,"Question text:";QTEXT$(Q,2)
730 PRINT #1,"Answer:";N;", ";QTEXT$(Q,N+2)
740  '  Find if answer is right or wrong and if unexpectedly so.
750 IF N=VAL(QTEXT$(Q,8)) THEN I=1:TEXT$="RIGHT" ELSE I=-1:TEXT$="WRONG"
760 IF (ABILITY-QDIFF(Q))*I<-2 THEN TEXT$="SURPRISINGLY "+TEXT$
770 PRINT #1,"This answer is: "+TEXT$:PRINT #1,""  'blank line after
780 PRINT QTEXT$(Q,1),FNU(QDIFF(Q)),N,TEXT$:NEXT P:CLOSE #1
790 PRINT NAM$;:GOSUB 610:RETURN  'Display estimated ability
```

```
800  '  This routine checks for Yes/No answers - no Enter key required
810 IF LEN(MSG$)<61 THEN PRINT MSG$; ELSE PRINT MSG$
820 PRINT "  Yes or No (Y/N):";:LOCATE ,,1
830 YESNO$=CHR$(ASC(INKEY$+" ") AND 223):WHILE YESNO$<>"Y" AND YESNO$<>"N"
840 YESNO$=CHR$(ASC(INKEY$+" ") AND 223):WEND:PRINT YESNO$:RETURN
850  '  Load the question file (9 lines per question +blank) into an array
860 QCOUNT=0:N=0:OPEN QFIL$ FOR INPUT AS #1:WHILE NOT EOF(1)
870 QCOUNT=QCOUNT+1:FOR I=1 TO 10:IF EOF(1) THEN 890
880 N=N+1:LINE INPUT #1,QTEXT$(QCOUNT,I):NEXT I
890 IF VAL(QTEXT$(QCOUNT,1))=<VAL(QTEXT$(QCOUNT-1,1)) THEN 930  'Check ID
900 I=VAL(QTEXT$(QCOUNT,8)):IF I<1 OR I>5 THEN 930  'Check correct answer
910 I=FNL(QTEXT$(QCOUNT,9)):IF I<FNL("1") OR I>FNL("200") THEN 930  'Units ok?
920 QDIFF(QCOUNT)=I:WEND:CLOSE#1:IF QCOUNT>0 THEN RETURN  'if all ok
930 PRINT "Error in question file, "+QFIL$+", at or before line ";N
940 PRINT "Test session ended":GOTO 170
950  '  Reestimation routine for question and test-taker measurements
960 PRINT "Reading test-takers' answers..."
970 PASKED=0:OPEN TFIL$ FOR INPUT AS #2:WHILE NOT EOF(2)
980 LINE INPUT #2,TEXT$:IF INSTR(TEXT$,"Test-taker")=0 THEN 1030
990  '  We have another test-taker - set his responses to unknown
1000 PASKED=PASKED+1:FOR Q=1 TO QCOUNT:RESULT(PASKED,Q)=-1:NEXT Q
1010 PABILITY(PASKED)=0:GOTO 1110
1020  '  Read previous estimate of test-taker's ability
1030 I=INSTR(TEXT$,"ability"):IF I=0 OR PABILITY(PASKED)>0 THEN 1050
1040 PABILITY(PASKED)=FNL(MID$(TEXT$,I+8)):GOTO 1110
1050 I=INSTR(TEXT$,"identifier"):IF I=0 THEN 1090  'is this a question id?
1060 Q=VAL(MID$(TEXT$,I+7))  'Question identifier - look up in table
1070 FOR I=1 TO QCOUNT:IF Q=VAL(QTEXT$(I,1)) THEN Q=I:GOTO 1110
1080 NEXT I:Q=0:GOTO 1110  'if not found flag as zero which is unused
1090 IF INSTR(TEXT$,"RIGHT")>0 THEN RESULT(PASKED,Q)=1:GOTO 1110  'save answer
1100 IF INSTR(TEXT$,"WRONG")>0 THEN RESULT(PASKED,Q)=0  '1=right 0=wrong
1110 WEND:CLOSE #2
1120 PRINT "Totalling scores...":QTOTAL=0:PTOTAL=0:RECOUNT=0
1130 FOR Q=1 TO QCOUNT:QASKED(Q)=0:QSCORE(Q)=0:NEXT Q
1140 FOR P=1 TO PASKED:PRESULT=0:PSCORE(P)=0:FOR Q=1 TO QCOUNT
1150     N=RESULT(P,Q):IF N<0 THEN 1180
1160     PRESULT=PRESULT+1:QASKED(Q)=QASKED(Q)+1
1170     PSCORE(P)=PSCORE(P)+N:QSCORE(Q)=QSCORE(Q)+N
1180   NEXT Q:IF PRESULT=0 THEN 1210
1190   IF PSCORE(P)>0 AND PSCORE(P)<PRESULT THEN PTOTAL=PTOTAL+1:GOTO 1210
```

```
1200   RECOUNT=1:FOR Q=1 TO QCOUNT:RESULT(P,Q)=-1:NEXT Q
1210 NEXT P:FOR Q=1 TO QCOUNT:IF QASKED(Q)=0 THEN 1240
1220  IF QSCORE(Q)>0 AND QSCORE(Q)<QASKED(Q) THEN QTOTAL=QTOTAL+1:GOTO 1240
1230  RECOUNT=1:FOR P=1 TO PASKED:RESULT(P,Q)=-1:NEXT P
1240 NEXT Q
1250 IF PTOTAL<2 OR QTOTAL<2 THEN PRINT "Not enough data to reestimate":RETURN
1260 IF RECOUNT=1 THEN 1120
1270 BIAS=(QTOTAL-1)*(PTOTAL-1)/(QTOTAL*PTOTAL)  'allow for statistical bias
1280 FOR Q=1 TO QCOUNT:IF QASKED(Q)<>0 THEN QDIFF(Q)=FNL(QTEXT$(Q,9))/BIAS
1290 NEXT Q:PADJ=0:FOR P=1 TO PASKED
1300 IF PSCORE(P)>0 THEN PABILITY(P)=PABILITY(P)/BIAS:PADJ=PABILITY(P)+PADJ
1310 NEXT P  'Sum current abilities to determine average ability level
1320  '  Now perform reestimation for 10 iterations.
1330 PRINT "Reestimating for";PTOTAL;"test-takers and";QTOTAL;"questions"
1340 FOR RECOUNT=1 TO 10:PRINT "Estimation cycle no.";RECOUNT
1350 PSUM=0:FOR Q=1 TO QCOUNT:QEXP(Q)=0:QVAR(Q)=0:NEXT Q
1360 FOR P=1 TO PASKED:IF PSCORE(P)=0 THEN 1470
1370  PEXP=0:PVAR=0:FOR Q=1 TO QCOUNT:IF QASKED(Q)=0 THEN 1420
1380   IF RESULT(P,Q)=-1 THEN 1420  'Look at each valid answer
1390   SUCCESS=1/(1+EXP(QDIFF(Q)-PABILITY(P)))  'Probability of success
1400   QEXP(Q)=QEXP(Q)+SUCCESS:PEXP=PEXP+SUCCESS  'Accumulate estimated scores
1410   N=SUCCESS*(1-SUCCESS):QVAR(Q)=QVAR(Q)+N:PVAR=PVAR+N  'sum variance
1420  NEXT Q
1430  RESIDUAL=PSCORE(P)-PEXP  'difference between actual and estimated
1440  IF PVAR>1 THEN RESIDUAL=RESIDUAL/PVAR  'amount to adjust by
1450  PABILITY(P)=PABILITY(P)+RESIDUAL  'new ability estimate
1460  PSE(P)=1/SQR(PVAR):PSUM=PSUM+PABILITY(P)  'standard error + ability sum
1470 NEXT P:PSUM=(PSUM-PADJ)/PTOTAL  'What is change in mean ability?
1480 FOR P=1 TO PASKED:IF PSCORE(P)>0 THEN PABILITY(P)=PABILITY(P)-PSUM
1490 NEXT P  'Keep mean ability of test-takers constant
1500 FOR Q=1 TO QCOUNT:IF QASKED(Q)=0 THEN 1540  'reestimate questions
1510  RESIDUAL=QSCORE(Q)-QEXP(Q)  'difference between actual and estimated
1520  IF QVAR(Q)>1 THEN RESIDUAL=RESIDUAL/QVAR(Q)  'amount to adjust by
1530  QDIFF(Q)=QDIFF(Q)-RESIDUAL  'new question difficulty estimate
1540 NEXT Q:NEXT RECOUNT:PRINT "Reestimation complete."
1550 INPUT "What is the name of the updated question file";QFIL$
1560 OPEN QFIL$ FOR OUTPUT AS #1:FOR Q=1 TO QCOUNT  '  write out all questions
1570 FOR I=1 TO 8:PRINT #1,QTEXT$(Q,I):NEXT I:IF QASKED(Q)=0 THEN 1600
1580 I=QDIFF(Q)*BIAS:QDIFF(Q)=I:SE=BIAS/SQR(QVAR(Q))  '  new difficulties
1590 PRINT #1,FNU(I);",";FNU(I-SE);"-";FNU(I+SE);",";  '  insert in line 9
```

```
1600 PRINT #1,QTEXT$(Q,9):PRINT #1,"":NEXT Q:CLOSE #1  'Append old estimate
1610  '  Now rewrite the test-taker file with revised abilities
1620 INPUT "What is the name of the revised test-taker file";TREVFIL$
1630 IF TFIL$=TREVFIL$ THEN 1620  'must be a different file
1640 OPEN TREVFIL$ FOR OUTPUT AS #1:PASKED=0  'read previous test-taker file
1650 OPEN TFIL$ FOR INPUT AS #2  'output revised test-taker file
1660 WHILE NOT EOF(2):LINE INPUT #2,TEXT$:PRINT #1,TEXT$  'copy over
1670 IF INSTR(TEXT$,"Test-taker")=0 THEN 1720  'is this next test-taker ?
1680 PASKED=PASKED+1:IF PSCORE(PASKED)=0 THEN 1720  'is his ability revised?
1690 ABILITY=PABILITY(PASKED)*BIAS:SE=PSE(PASKED)*BIAS  'remove bias
1700 PRINT #1,"Revised estimated ability:";FNU(ABILITY)
1710 PRINT #1,"Probable ability range:";FNU(ABILITY-SE);"-";FNU(ABILITY+SE)
1720 WEND:CLOSE #2:CLOSE #1:TFIL$=TREVFIL$:RETURN  'Use new test-taker file
```

Figure 6. BASIC listing of program to administer and score a computer-adaptive test containing multiple-choice questions. It can also reestimate the difficulty of the questions to aid in improving the effectiveness of the test.

```
Question identifier: 2

Please select the correct answer to the following question:

Which country is in the continent of Africa?

The answer is one of:

 1 . Australia
 2 . Bolivia
 3 . Cambodia
 4 . Nigeria
 5 . Romania

Type the number of your selection here:
```

Figure 1. The computer chooses and displays a multiple-choice question of the appropriate level of difficulty.

Summary report on questions administered to Fred

Identifier	Difficulty	Answer	Right/Wrong
2	96	4	RIGHT
24	99	2	RIGHT
1	104	3	WRONG
25	114	5	RIGHT
7	106	2	WRONG
13	111	4	RIGHT
12	105	2	RIGHT
15	109	1	WRONG
3	85	2	SURPRISINGLY WRONG
18	103	3	RIGHT

Fred scored in the range from 101 to 115 at about 108 after 10 questions

Figure 2. At the end of the test session a summary of the test session is displayed, as well as an estimate of the test-taker's ability.

```
1
Which city is the capital of West Germany ?
Berlin
Bonn
Dortmund
Hamburg
Weimar
2
104

2
Which country is in the continent of Africa?
Australia
Bolivia
Cambodia
Nigeria
Romania
4
96

7
Which city is the state capital of N..    k?
Albany
Buffalo
New York City
Rochester
Syracuse
1
106
```

Figure 3. Example of questions entered on the question file using a text editor. Each question has an identifying number (in ascending order but gaps are allowed), then the text of the question, the 5 possible answers, the number of the correct answer (1-5), and a preliminary estimate of the questions difficulty, relative to 100.

```
Test-taker's name: George
Estimated ability: 108
Probable ability range: 101 - 115

Question identifier: 2
Estimated difficulty: 96
Question text:Which country is in the continent of Africa?
Answer: 1 , Australia
This answer is: WRONG

Question identifier: 24
Estimated difficulty: 99
Question text:Which country has no sea coast?
Answer: 4 , Switzerland
This answer is: RIGHT
```

Figure 4. Details of each test session are written to the test-taker file on disk. The include the test-taker's name and estimated ability, and the range in which it probably lies. Then each question he was asked and how he answered it.

```
1
Which city is the capital of West Germany ?
Berlin
Bonn
Dortmund
Hamburg
Weimar
2
106, 99 - 113, 104
```

Figure 5. Your question file, with the reestimated difficulty and range inserted before your difficulty estimate on the ninth line of the question.

```
Test-taker's name: George
Revised estimated ability: 106
Probable ability range: 100 - 112
Estimated ability: 108
Probable ability range: 101 - 115

Question identifier: 2
Estimated difficulty: 96
Question text:Which country is in the continent of Africa?
Answer: 1 , Australia
This answer is: WRONG
```

Figure 6. The test-taker file, showing a revised ability estimate included before the ability estimate made at the time of the interactive test.

Explanation of BASIC variable names

Variable	Description
ABILITY	Estimate of ability, sometimes assumes next answer wrong
ABILRIGHT	Estimate of ability, if next answer right
ACCURACY	Maximum width of likely zone of estimate, =2*standard error
BIAS	Adjustment for statistical bias
FNL	Function to convert external units to logits (-10 to +10)
FNU	Function to convert logits to external units (1-200)
I	Subscript index and numerical working variable
MSG$	Message to be sent to screen
N	Numerical working variable
NAM$	Name of test taker
P	Current location in test-taker array
PABILITY(50)	Estimated test-takers' abilities
PADJ	Total of previous ability estimates
PANSWER(50)	Answers keyed in by test-taker to questions asked
PASKED	Number of questions asked or number of answers to a question
PEXP	Expected score by test-taker
PQUESTION(50)	Location in QTEXT$ array of questions asked test-taker
PRESULT	Count of questions asked a test-taker
PSCORE(50)	Count of test-taker's correct answers, i.e. raw score
PSE(50)	Standard error of estimation (accuracy) of ability estimates
PSUM	Sum of all ability estimates
PTOTAL	Total number of test-takers being reestimated

PVAR	Variance of expected score for a test-taker
Q	Location in question arrays
QASKED(50)	Number of times question has been asked
QCOUNT	Number of questions in question file
QDIFF(50)	Difficulty estimates for questions
QEXP(50)	Estimated score based on ability and difficulty estimates
QFIL$	Name of question text file
QSCORE(50)	Count of number of correct answers to each question
QSELECT	Location of next question to be displayed in question array
QTEXT$(50,10)	Text of questions and answer options (10 lines per question)
QTOTAL	Total number of questions being reestimated
QVAR(50)	Variance of expected score by test-taker's on a question
RECOUNT	Working variable to control recounting and reestimating
RESIDUAL	Difference between actual and estimated scores
RESULT(50,50)	Answers by test-takers to questions: 1=correct 0=incorrect -1=unknown (not taken)
SE	Standard error of estimation of ability measure
SUCCESS	Probability of correct answer by test-taker to question
TEXT$	Text of line from an input file
TFIL$	Name of file of test-taker's abilities and responses
TREVFIL$	Name of revised test-taker file
YESNO$	Response from keyboard to displayed question: Y=yes N=no

Figure 7. Names and usage of variables used in the BASIC program

Arrays are dimensioned for 50 test-takers and questions.

The arrays can be made smaller or larger but should all be the same size.